LLM-Systems Basics

EECS 598

Jiachen Liu

2024/1

SymbioticLab

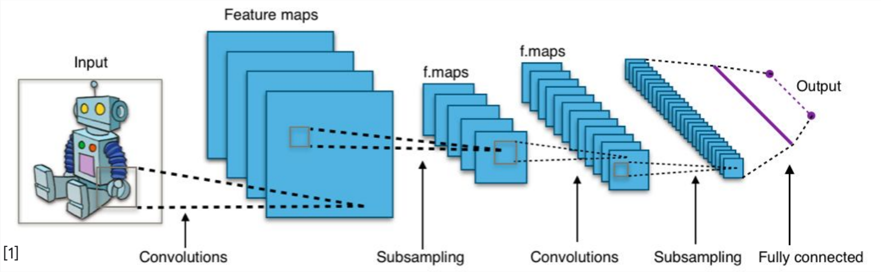
UNIVERSITY OF MICHIGAN

Agenda

1. Why Transformer and What is Transformer ?
2. Why LLM is unique in terms of System Design?
3. How can we better improve the system performance of LLM ?

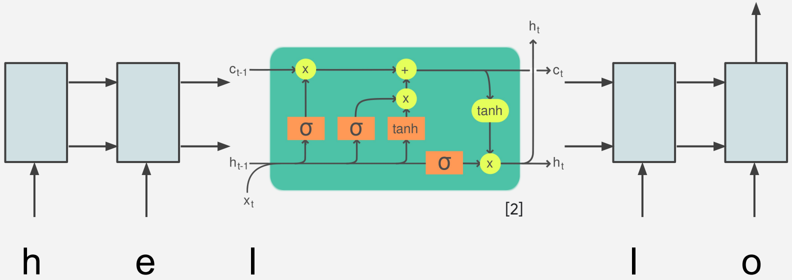
Computer Vision

Convolutional NNs (+ResNets)



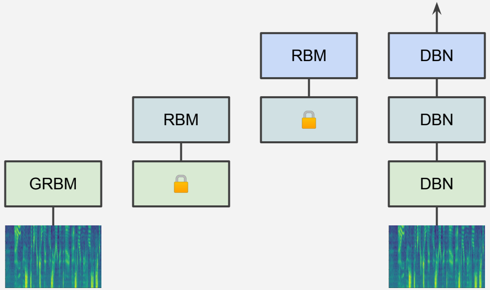
Natural Lang. Proc.

Recurrent NNs (e.g. LSTMs)



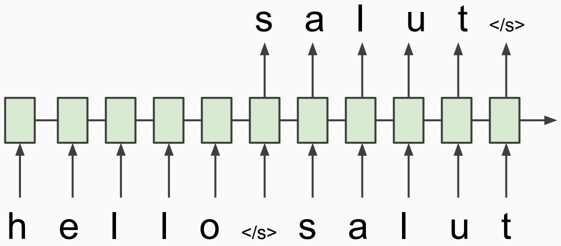
Speech

Deep Belief Nets (+non-DL)



Translation

Seq2Seq



RL

BC/GAIL

Algorithm 1 Generative adversarial imitation learning

1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters θ_0, w_0
2: **for** $i = 0, 1, 2, \dots$ **do**
3: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
4: Update the discriminator parameters from w_i to w_{i+1} with the gradient

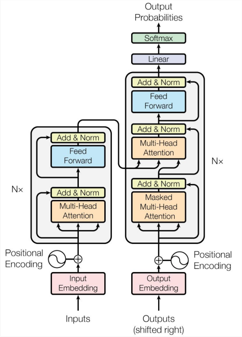
$$\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))] \qquad (17)$$

5: Take a policy step from θ_i to θ_{i+1}, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$. Specifically, take a KL-constrained natural gradient step with

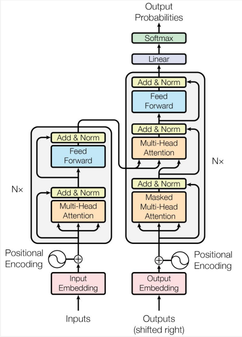
$$\hat{\mathbb{E}}_{\tau_i}[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)] - \lambda \nabla_\theta H(\pi_\theta),$$
$$\text{where } Q(\bar{s}, \bar{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) \mid s_0 = \bar{s}, a_0 = \bar{a}] \qquad (18)$$

6: **end for**

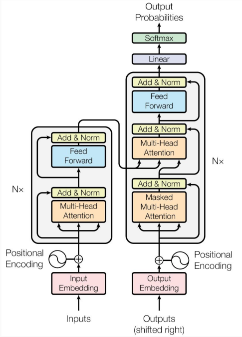
Computer Vision



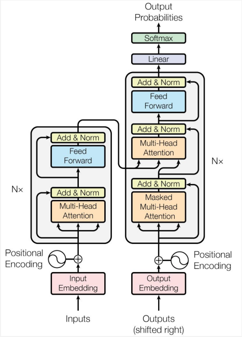
Natural Lang. Proc.



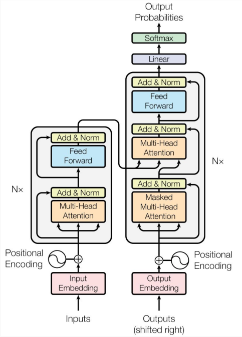
Reinf. Learning



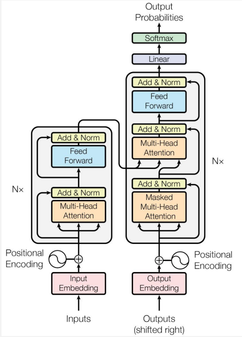
Speech



Translation



Graphs/Science



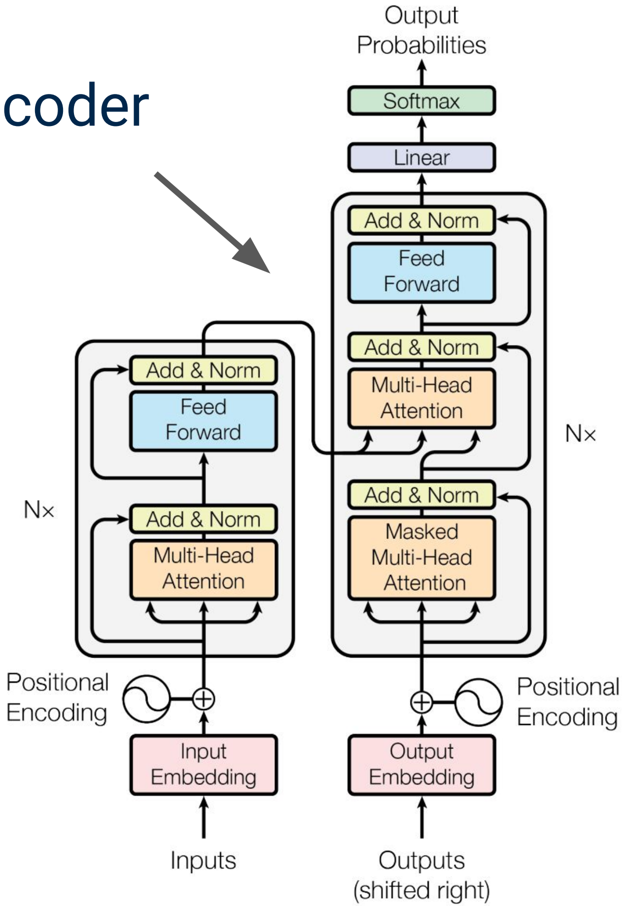
Transformer image source: "Attention Is All You Need" paper

What’s happening inside Transformers?

Encoder-Decoder

Encoder

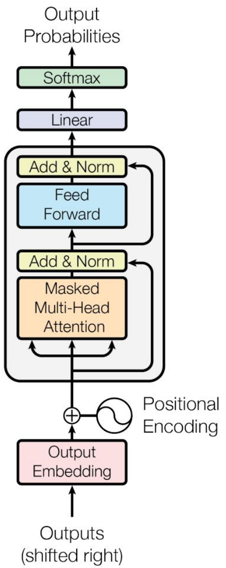
Decoder



Attention is all you need

Decoder-only
GPT

[sat_]

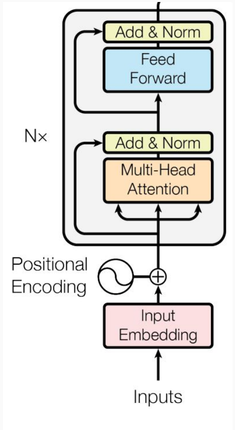


[START] [The_] [cat_]

Attention is all you need

Encoder-only
BERT

[*] [*] **[sat_]** [*] **[the_]** [*]



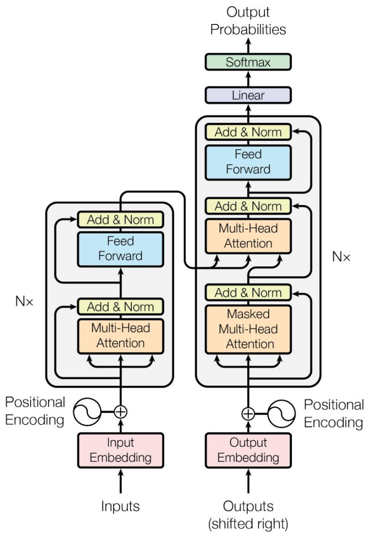
[The_] [cat_] **[MASK]** [on_] **[MASK]** [mat_]

Enc-Dec
T5

Das ist gut.

A storm in Attala caused 6 victims.

This is not toxic.

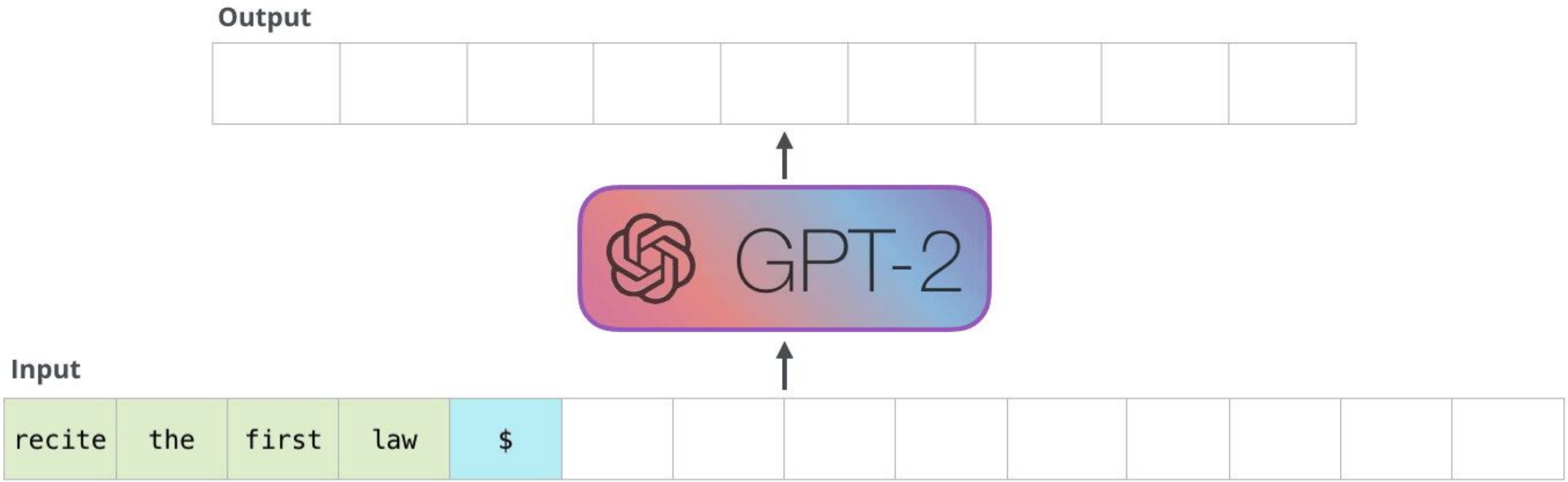


Translate EN-DE: This is good.

Summarize: state authorities dispatched…

Is this toxic: You look beautiful today!

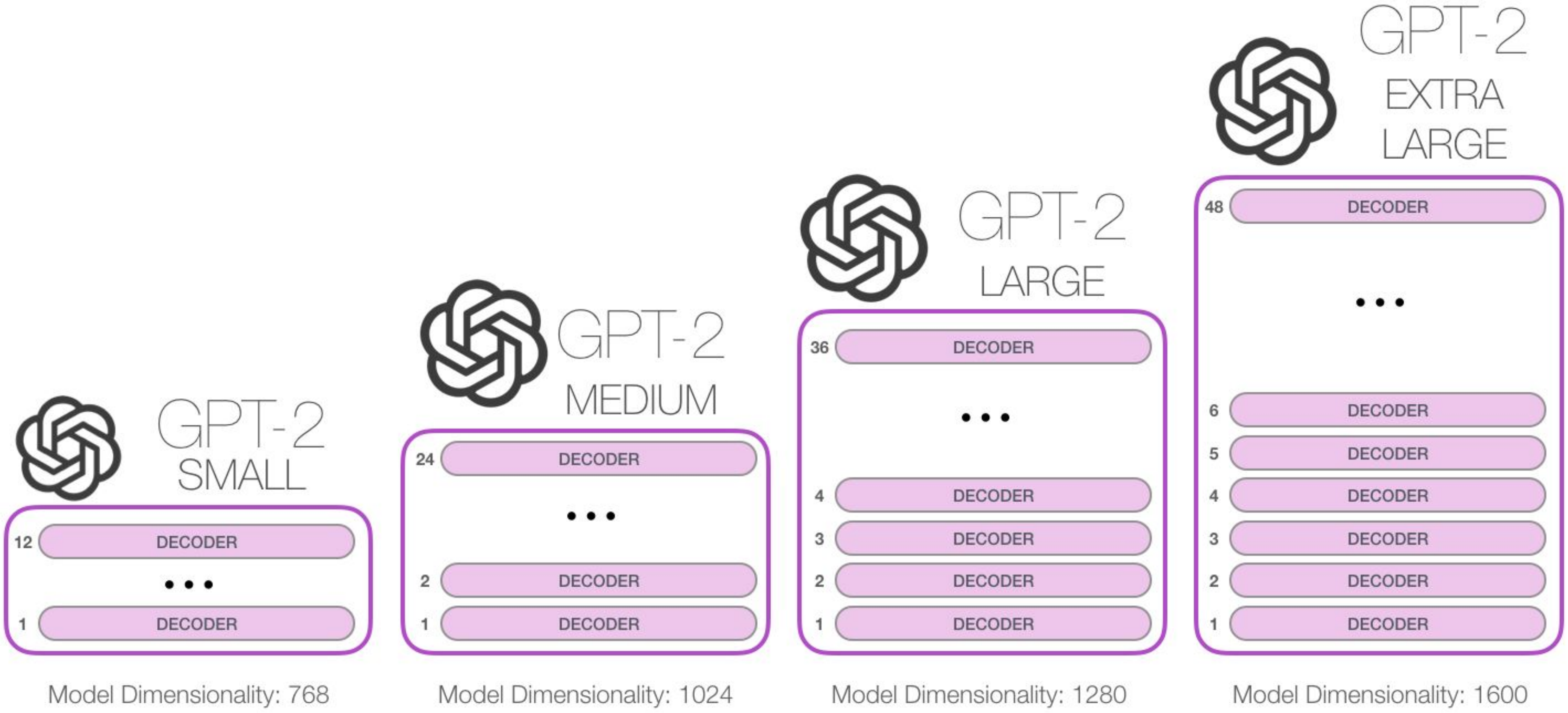
High-level Workflow of GPT Inference



Auto-Regressive Decoding

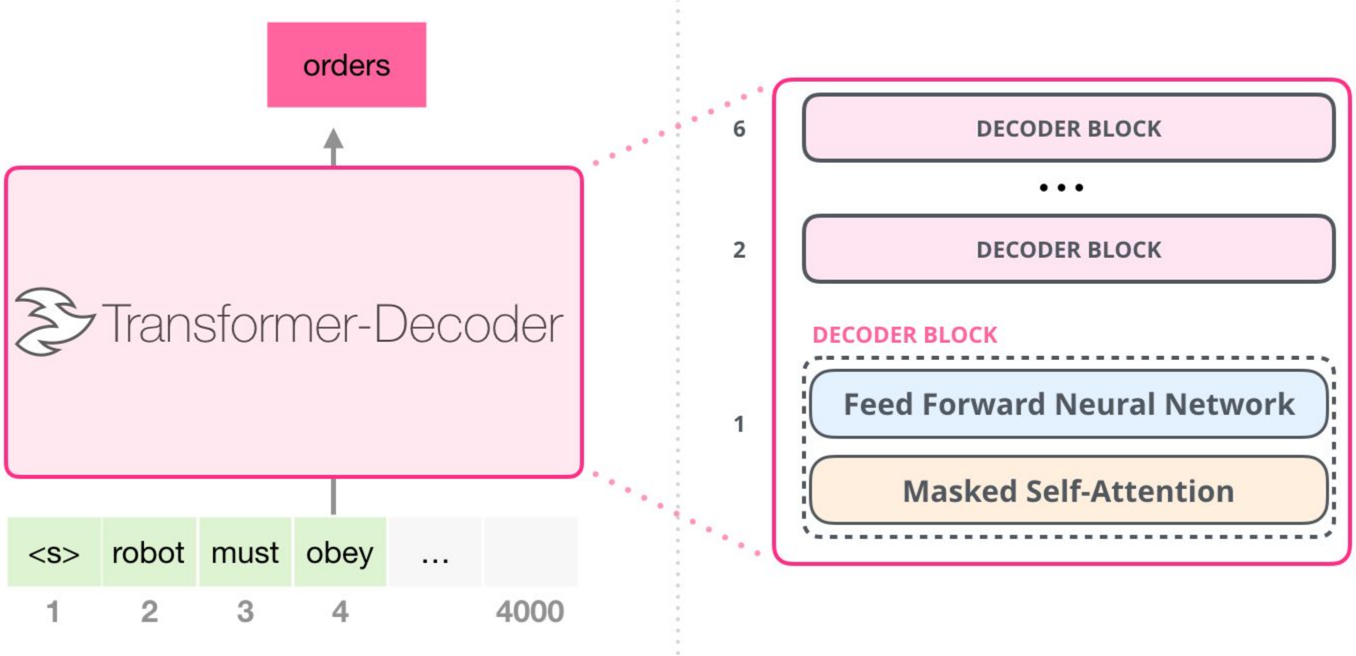
https://jalammar.github.io/illustrated-gpt2/

Zoom in GPT



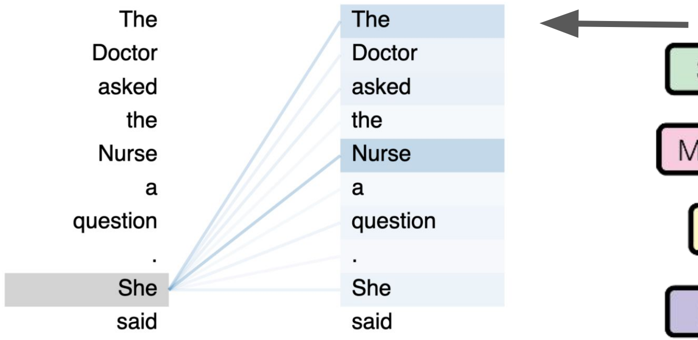
https://jalammar.github.io/illustrated-gpt2/

Zoom in the Decoder Layer

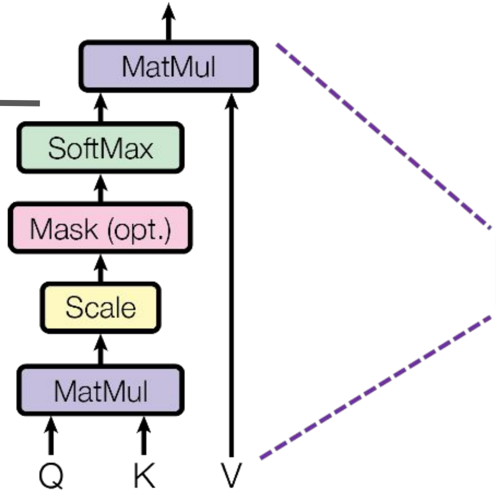


https://jalammar.github.io/illustrated-gpt2/

Zoom in the Masked Self-Attention

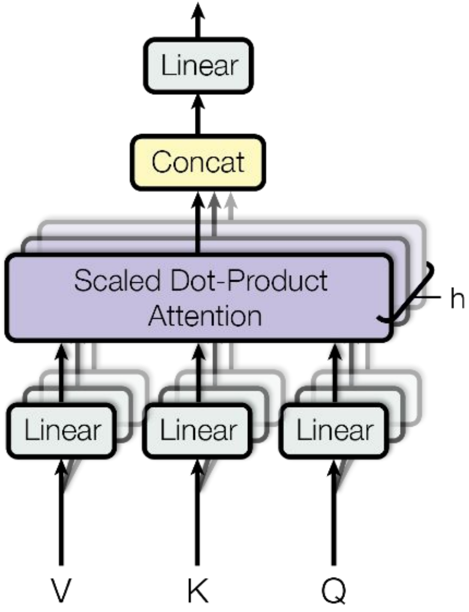


Scaled Dot-Product Attention



$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Multi-Head Attention

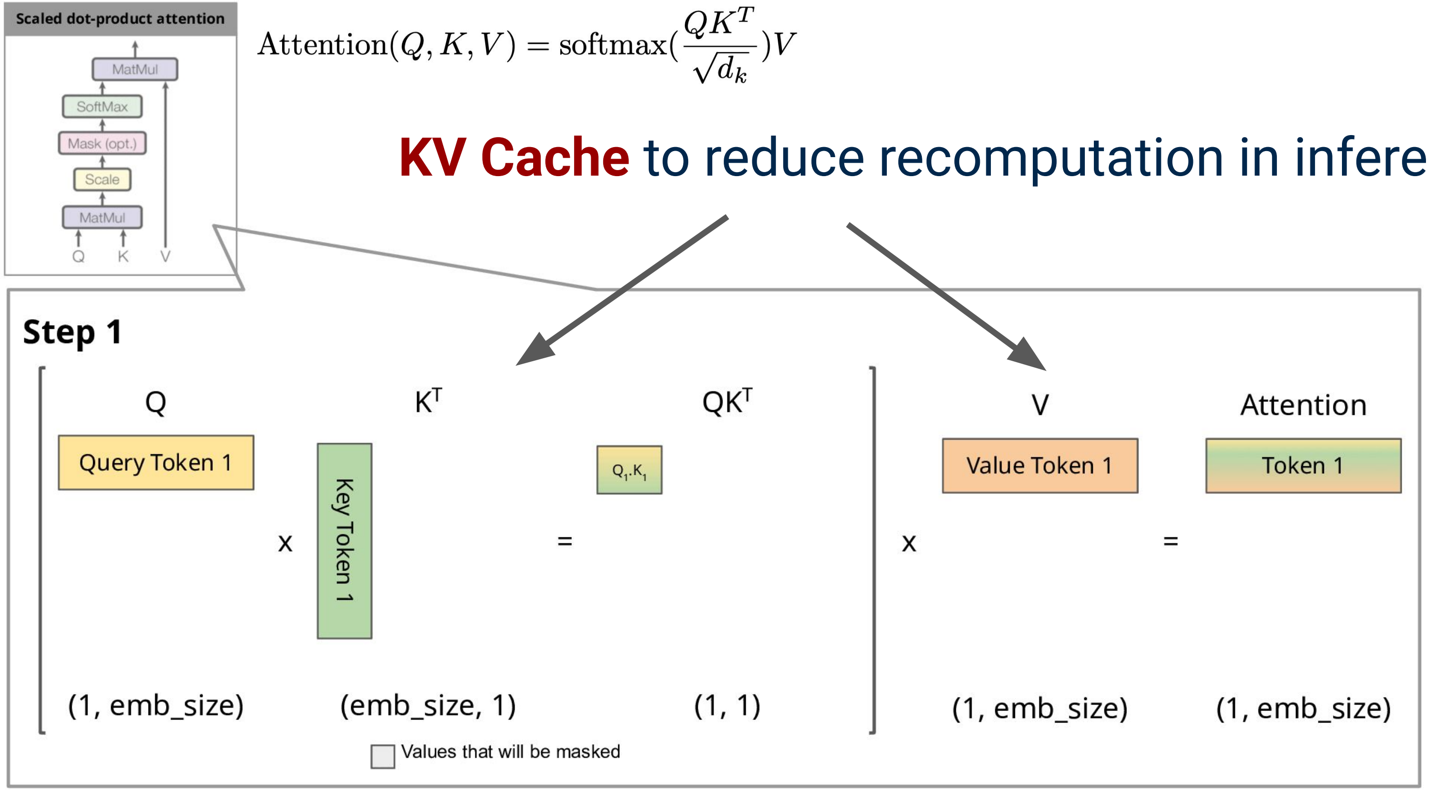


$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Zoom in Scaled Dot Product Attention

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

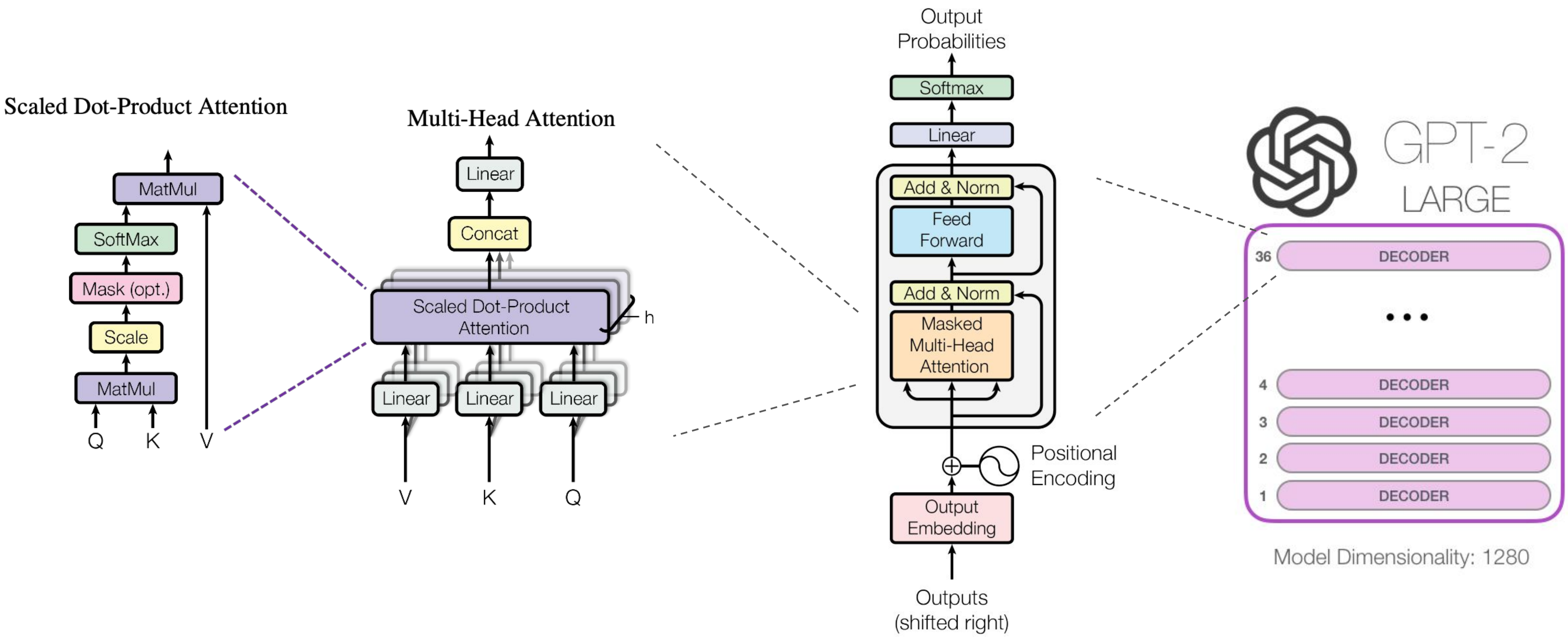
KV Cache to reduce recomputation in inference



Zoom-in! (simplified without Scale and Softmax)

https://medium.com/@joaolages/kv-caching-explained-276520203249

Recap: Transformer Architecture



How did the Deep Learning community end up with Transformers?

Advantage

- Highly Paralizable / Scalable
- Long Term Memory

Disadvantage

- Sensitive to Data Quality and Quantity
- High Computational Demand

Computation Resources Needed For LLMs

Training

- Model: GPT-3 175 B.
  - 350GB model weight for half-precision training
  - 350GB activation, gradient, per optimizer state
- Resources: V100 w/ 32GB GPU memory
- Time: **355 GPU years** over 300 Billion tokens
- Cost: > **$4.6M** for V100

Inference

- Model: GPT-3 175 B.
  - 350GB model weight
  - ~700 GB KV Cache for 100 concurrent sequences with context length 1000
  - Energy consumption

https://lambdalabs.com/blog/demystifying-gpt-3

Characteristics of Transformers - Training

1. Huge Model Size
   a. Need efficient parallelism for computation and communication
2. Long training time
   a. Need efficient fault tolerance systems

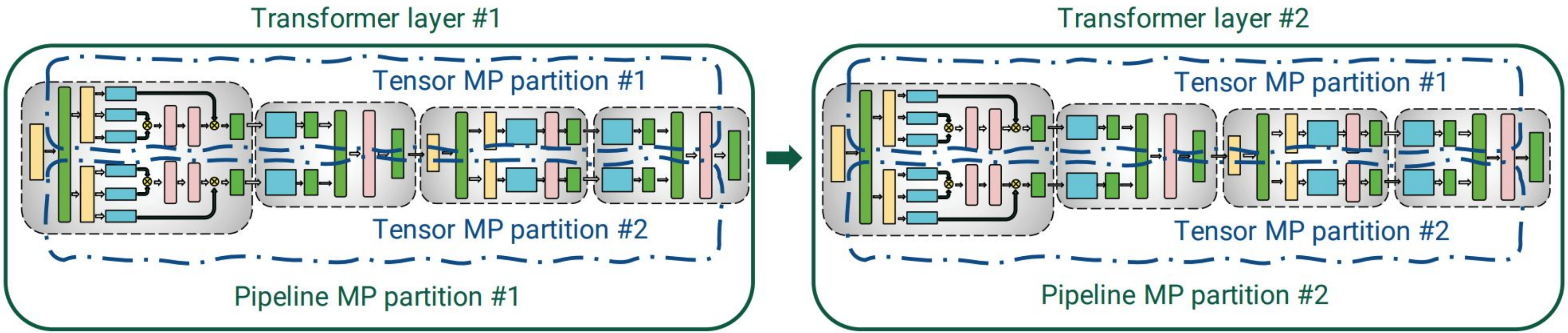
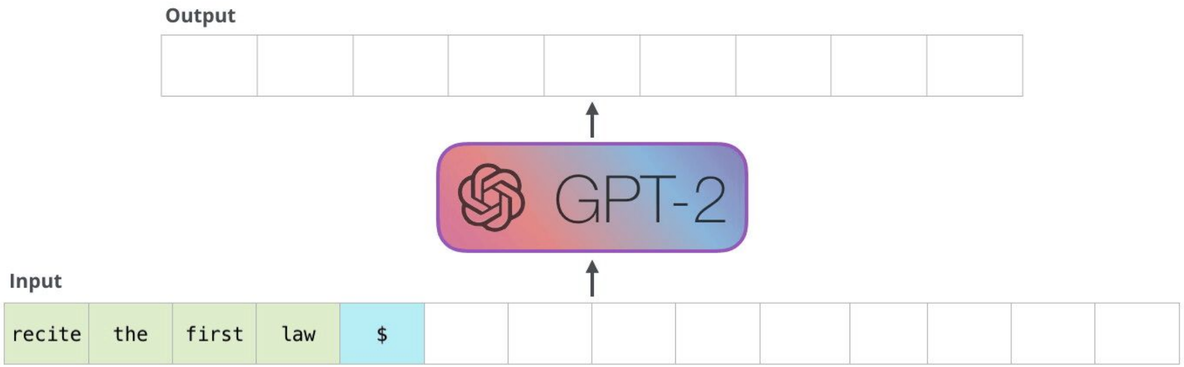


Figure 2: Combination of tensor and pipeline model parallelism (MP) used in this work for transformer-based models.

Efficient Large-Scale Language Model Training on GPU Clusters Using Megatron-LM

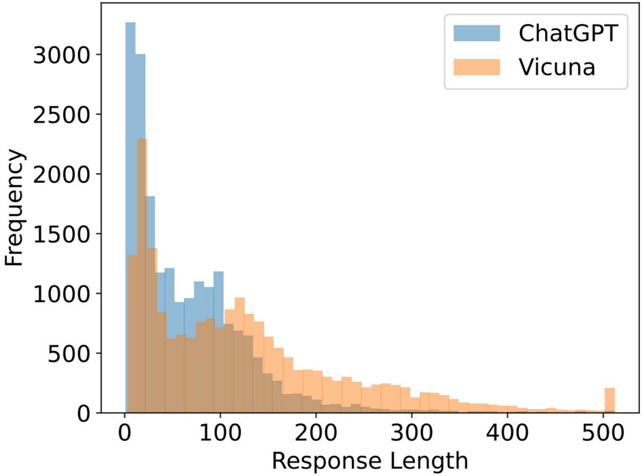
Characteristics of Transformers - Inference

1. **Auto-regressive** decoding
   - The model generates one token at a time, taking into account previous token



Characteristics of Transformers - Inference

2. **Unknown response length** → Unknown inference latency

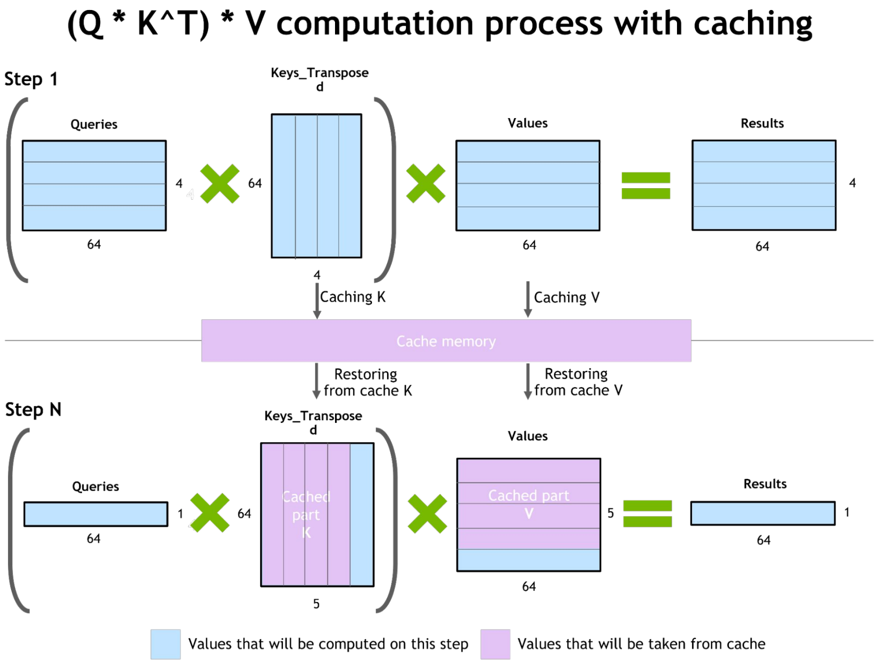


(a) Response length distribution of 10k instructions from ChatGPT and Vicuna. Response lengths larger than 512 are truncated.

Response Length Perception and Sequence Scheduling: An LLM-Empowered LLM Inference Pipeline

Characteristics of Transformers - Inference

3. **Different computing phase**
   - Prefill: digest prompt
   - Decode: predict next token

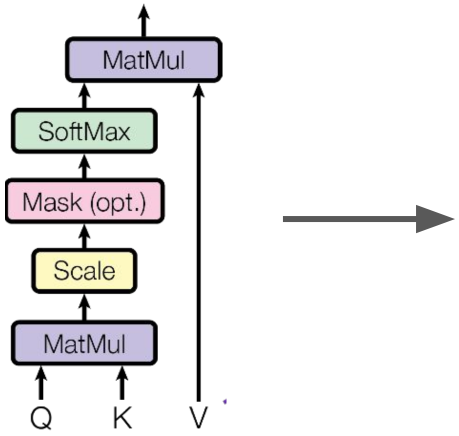


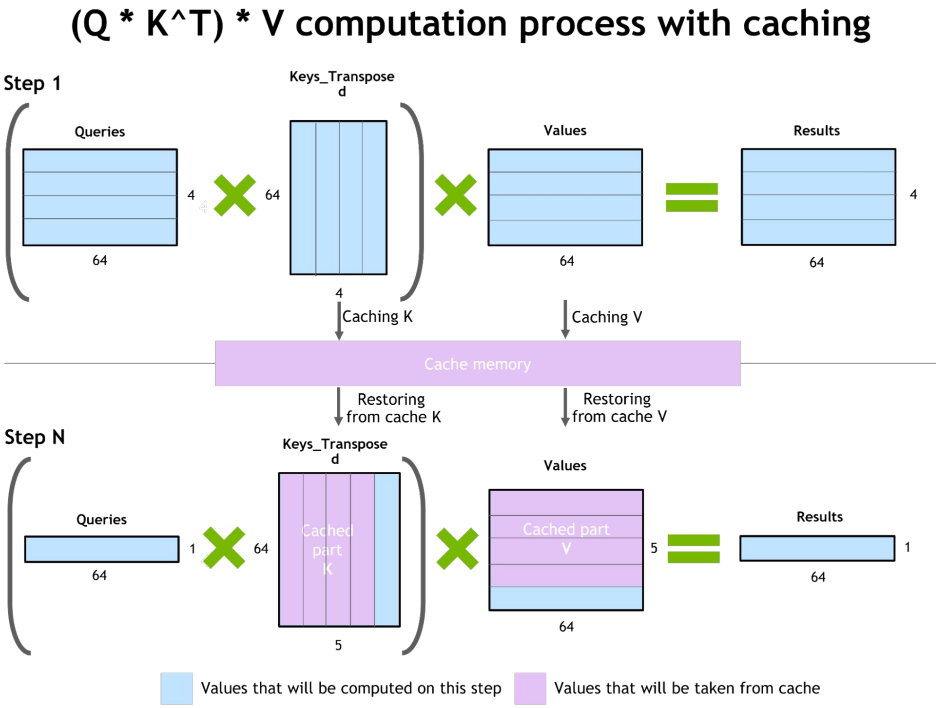
https://developer.nvidia.com/blog/accelerated-inference-for-large-transformer-models-using-nvidia-fastertransformer-and-nvidia-triton-inference-server/

Characteristics of Transformers - Inference

4. **Memory capacity intensive** due to KV Cache

- Long context sequence



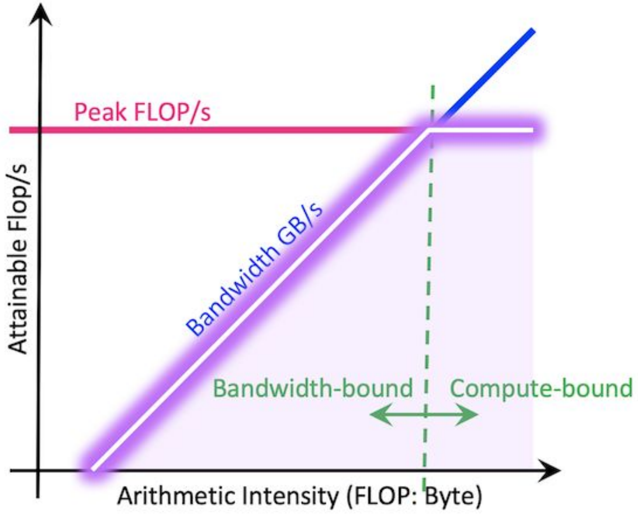


https://developer.nvidia.com/blog/accelerated-inference-for-large-transformer-models-using-nvidia-fastertransformer-and-nvidia-triton-inference-server/

Characteristics of Transformers - Inference

5. Memory IO Bounded

- GPU's performance is limited by the speed at which it can read from or write to its memory

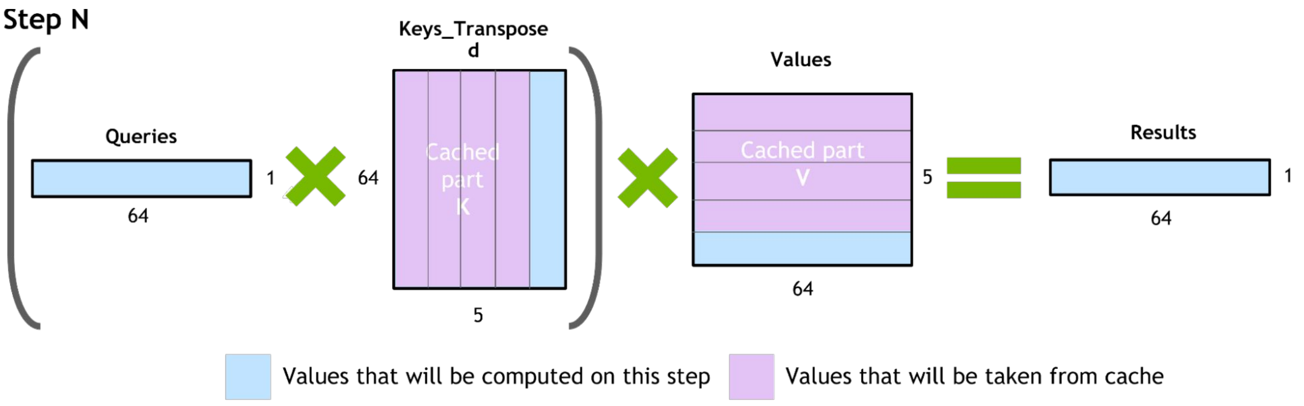


GPU	Mem Bandwidth (GB/s)	FLOPs (Tensor)	Arithmetic Intensity
V100	~900 (HBM2)	125 TFLOPS	~140
A100	~1555 (HBM2)	312 TFLOPS	~200

Characteristics of Transformers - Inference

5. Memory IO Bounded

- GPU's performance is limited by the speed at which it can read from or write to its memory
- Arithmetic intensity for
  - Vector - matrix multiplication in Attention layer ≈ 2
  - Matmul in FFN layer ≈ 2 * Batch Size

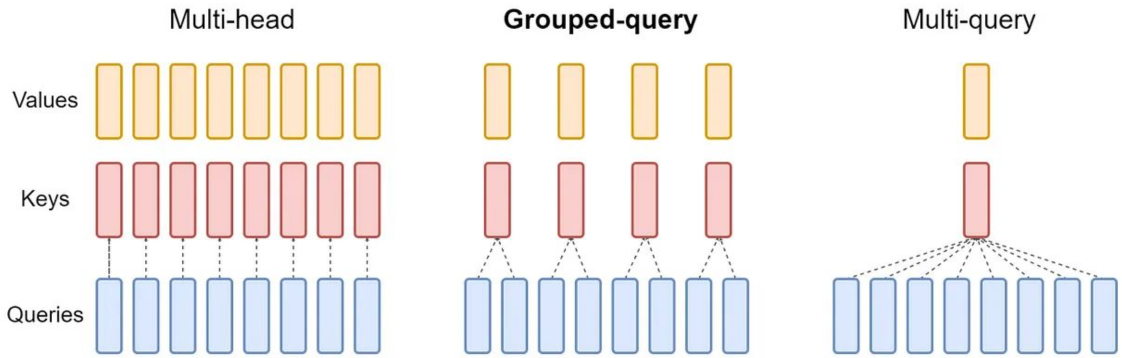


https://developer.nvidia.com/blog/accelerated-inference-for-large-transformer-models-using-nvidia-fastertransformer-and-nvidia-triton-inference-server/

Characteristics of Transformers - Inference

5. **Memory IO Bounded**

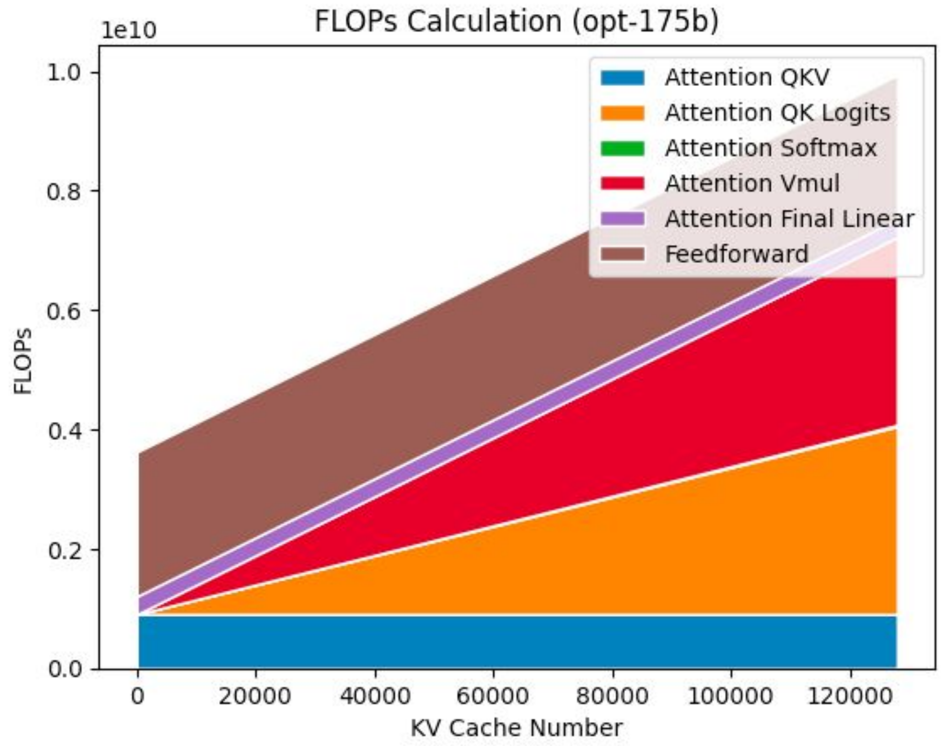
    - GPU's performance is limited by the speed at which it can read from or write to its memory
    - Arithmetic intensity for
        - Vector - matrix multiplication in Attention layer ≈ 2
        - Matmul in FFN layer ≈ 2 * Batch Size
    - Other attention structure with higher arithmetic intensity



https://developer.nvidia.com/blog/accelerated-inference-for-large-transformer-models-using-nvidia-fastertransformer-and-nvidia-triton-inference-server/

Observation of Transformers - Inference

6. Inference FLOPs **breakdown** – Batch size = 1



Efficient LLM Inference System Solutions

1. Compression, Quantization, Pruning
2. Parallel computation
3. Memory management
4. Request scheduling
5. Kernel optimization

Reference:

1. Efficient Large Language Models: A Survey
2. https://github.com/AmberLJC/LLMSys-PaperList/

Q & A